CLAIMS

1. A method for managing variable size memory partitioning over a plurality of memory buffer sizes, the method comprising:

receiving a first request for a memory buffer of a given size to store received data therein;

stealing the memory buffer by:

storing at least a portion of cached data in the memory buffer to a secondary memory; and

storing the received data in the memory buffer where the cached data was previously stored;

creating a buffer steal history record, wherein the buffer steal history record includes an unique identifier for identifying a location in the secondary memory of the cached data, wherein the buffer steal history record is used to determine whether a subsequent request for stealing the memory buffer is a re-access of the cached data previously written to the secondary memory.

2. The method of claim 1, wherein the creating a buffer steal history record includes creating the buffer steal buffer with a sequence number that is associated with a steal sequence of the memory buffer.

3. The method of claim 1, further comprising:

receiving the second request for the memory buffer to store additional received data; and

reaccessing the portion of cached data within the secondary memory using the buffer steal history record.

4. The method of claim 1, further comprising:

partitioning the memory buffer into at least one subgroup having similarly sized buffers grouped within.

5. The method of claim 4, wherein the creating a buffer steal history record includes creating the buffer steal buffer with a sequence number that is associated with a steal sequence of the memory buffer from the subgroup having similarly sized buffers.

6. The method of claim 5, wherein the partitioning the memory buffer into at least one subgroup having similarly sized buffers grouped within includes portioning the memory buffer based on the buffer steal history.

7. The method of claim 6, wherein the partitioning the memory buffer further comprises for each subgroup calculating a priority for the subgroup by:

accumulating an average difference for all buffers stolen and reaccessed from the subgroup;

determining an importance of reaccesses in the subgroup: and

identifying the memory buffer by comparing the priority of each subgroup, wherein the memory buffer is located within a subgroup having the lowest priority.

8. The method of claim 1, further comprising:

adding the buffer steal history record to a buffer steal history queue, wherein the buffer steal history queue comprises zero or more buffer steal history records.

9. A method for balancing multiple memory buffer sizes comprising:

building a buffer steal history queue comprising one or more buffer steal history records, wherein each of the one or more buffer steal history records includes a unique identifier for identifying one or more cached data stored in a secondary memory location;

receiving a request to store data in a buffer;

comparing the received request to each of the unique identifiers of each of the one or more buffer steal history records; and

reaccessing the cached data from the secondary memory when the data in the request matches one of the unique identifiers of a buffer steal history record.

10. The method of claim 9, wherein the buffer steal history record further comprises a sequence number, the method further comprising:

computing a difference between the sequence number of the buffer steal history record and a current sequence number for a corresponding buffer size subgroup;

comparing the difference to the corresponding buffer size subgroup; and

reaccessing the corresponding buffer size subgroup for the cached data when the corresponding buffer size subgroup is at least equal to the difference.

11. The method of claim 10, further comprising:

rereading the cached data from the secondary memory when the corresponding buffer size subgroup is smaller than the difference.

12. The method of claim 11, wherein the difference comprises the number of buffers stolen from the corresponding buffer size subgroup between a prior buffer steal and a current requested reaccess.

13. The method of claim 11, further comprising:

for each of a plurality of subgroups, wherein each subgroup comprises a plurality of similarly sized buffers, calculating a priority for the subgroup by:

accumulating an average difference for all buffers stolen and reaccessed from the subgroup;

determining an importance of reaccesses in the subgroup; and

identifying a buffer steal target in response to a request for a buffer by comparing the priority of each subgroup, wherein the buffer steal target is a buffer having the lowest priority.

14. A system for balancing multiple memory buffer sizes comprising:

a plurality of buffers arranged within a plurality of subgroups, wherein the subgroups contain one or more similarly sized buffers,

a buffer steal history queue comprising a plurality of buffer steal history records; and

a controller including a memory manager, wherein the memory manager is adapted to:

receive a first request for a buffer to store received data in the memory;

identify a buffer from one of the plurality of buffers, wherein the buffer contains cached data;

steal the buffer by:

storing the cached data in a secondary store location; and

storing the received data in the buffer;

create a buffer steal history record, wherein the buffer steal history record includes a unique identifier for identifying a location in the secondary store of the cached data, wherein the buffer steal history record is used to determine whether a subsequent request for stealing the memory buffer is a re-access of the cached data previously written to the secondary memory.

15. The system of claim 14, wherein the memory is partitioned into a plurality of subgroups having similarly sized buffers grouped within, wherein the buffer is grouped within one of the plurality of subgroups.

16. The system of claim 14, wherein the memory manager is further adapted to:

for each of the plurality of subgroups calculating a priority for the subgroup by:

accumulating an average difference for all buffers stolen and reaccessed from the subgroup;

determining an importance of reaccesses in the subgroup; and

identifying the buffer by comparing the priority of each subgroup, wherein the buffer is located within a subgroup having the lowest priority.

17. The system of claim 14, wherein the buffer steal history record is added to the buffer steal history queue.

18. A system for balancing multiple memory buffer sizes comprising:

a plurality of buffers arranged within a plurality of subgroups, wherein the subgroups contain one or more similarly sized buffers, and

a buffer steal history queue comprising a plurality of buffer steal history records, wherein each of the one or more buffer steal history records includes a unique identifier for identifying one or more cached data stored in a secondary memory location; and

a controller including a memory manager, wherein the memory manager is adapted to:

receive a request to store data in a buffer from one of the plurality of buffer steal history records;

compare the received request to each of the unique identifiers of each of the one or more buffer steal history records; and

reaccess the cached data from the secondary memory when the data in a subsequent request matches a unique identifier of a buffer steal history record.

19. The system of claim 18, wherein the buffer steal history record further comprises a sequence number, and wherein the memory manager is further adapted to:

compute a difference between a sequence number of the buffer steal history record and a current sequence number for a corresponding buffer size subgroup;

compare the difference to the corresponding buffer size subgroup; and

reaccess the corresponding buffer size subgroup for the cached data when the corresponding buffer size subgroup is at least the size of the difference.

20. The method of claim 14, for balancing multiple memory buffer sizes as defined in claim 16, wherein the memory manager is further adapted to:

reread the cached data from the secondary memory when the corresponding buffer size subgroup is smaller then the size of the difference.

21. The system of claim 14, wherein the unique identifier comprises the disk address of the buffer's data.

22. A computer program product for for managing variable size memory partitioning over a plurality of memory buffer sizes, the computer program product comprising:

a storage medium readable by a processing circuit and storing computer instructions for execution by the processing circuit for performing a method comprising:

receiving a first request for a memory buffer of a given size to store received data therein;

stealing the memory buffer by:

storing at least a portion of cached data in the memory buffer to a secondary memory; and

storing the received data in the memory buffer where the cached data was previously stored;

creating a buffer steal history record, wherein the buffer steal history record includes an unique identifier for identifying a location in the secondary memory of the cached data, wherein the buffer steal history record is used to determine whether a subsequent request for stealing the memory buffer is a re-access of the cached data previously written to the secondary memory.

23. The computer program product according to claim 22, wherein the creating a buffer steal history record includes creating the buffer steal buffer with a sequence number that is associated with a steal sequence of the memory buffer.

24. The computer program product according to claim 22, further comprising:

receiving the second request for the memory buffer to store additional received data; and

reaccessing the portion of cached data within the secondary memory using the buffer steal history record.

25. The computer program product according to claim 22, further comprising:

partitioning the memory buffer into at least one subgroup having similarly sized buffers grouped within.

26. The computer program product according to claim 25, wherein the creating a buffer steal history record includes creating the buffer steal buffer with a sequence number that is associated with a steal sequence of the memory buffer from the subgroup having similarly sized buffers.

27. The computer program product according to claim 26, wherein the partitioning the memory buffer into at least one subgroup having similarly sized buffers grouped within includes portioning the memory buffer based on the buffer steal history.

28. The computer program product according to claim 27, wherein the partitioning the memory buffer further comprises for each subgroup calculating a priority for the subgroup by:

accumulating an average difference for all buffers stolen and reaccessed from the subgroup;

determining an importance of reaccesses in the subgroup: and

identifying the memory buffer by comparing the priority of each subgroup, wherein the memory buffer is located within a subgroup having the lowest priority.

29. The computer program product according to claim 22, further comprising:

adding the buffer steal history record to a buffer steal history queue, wherein the buffer steal history queue comprises zero or more buffer steal history records.